UNIVERSIDAD EAFIT
ST1800/ST1801 ALMACENAMIENTO Y RECUPERACIÓN DE INFORMACIÓN
2025-1

Nombre:
Jose Luis Bedoya Martinez, jlbedoyam@eafit.edu.co
Álvaro Javier Mutis Guerrero, ajmutisg@eafit.edu.co
Juan Luis Amaya Arbeláez, jlamayaa@eafit.edu.co
Kevin Genez Valencia, kgenezv@eafit.edu.co

Enlace para acceder a toda la documentación en GitHub:

https://github.com/kdgenez/ARI-T1-P2.git

Contenido

Parte 1:

1. LABORATORIO 0:

Activar la cuenta y el curso en AWS ACADEMY

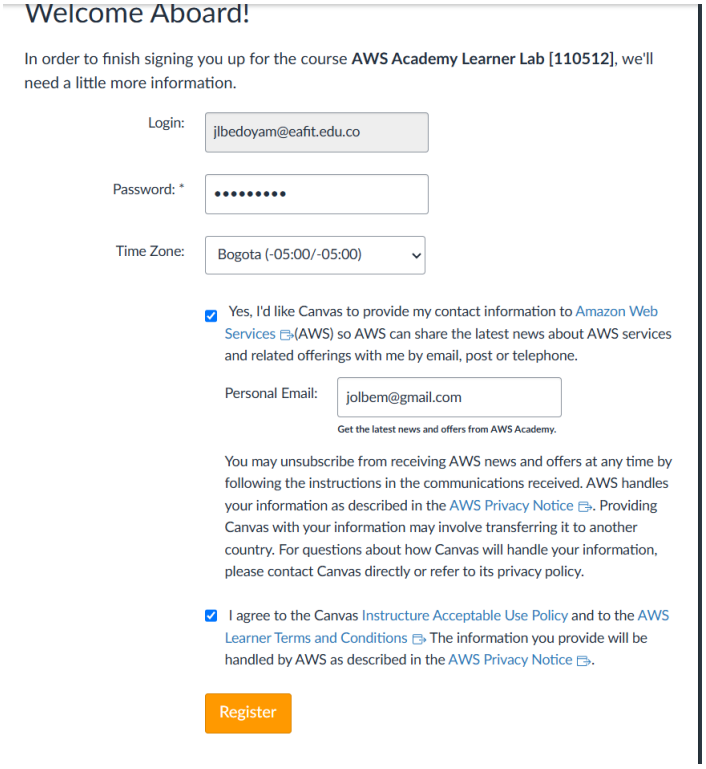


Ilustración 1 Activar la cuenta en AWS ACADEMY

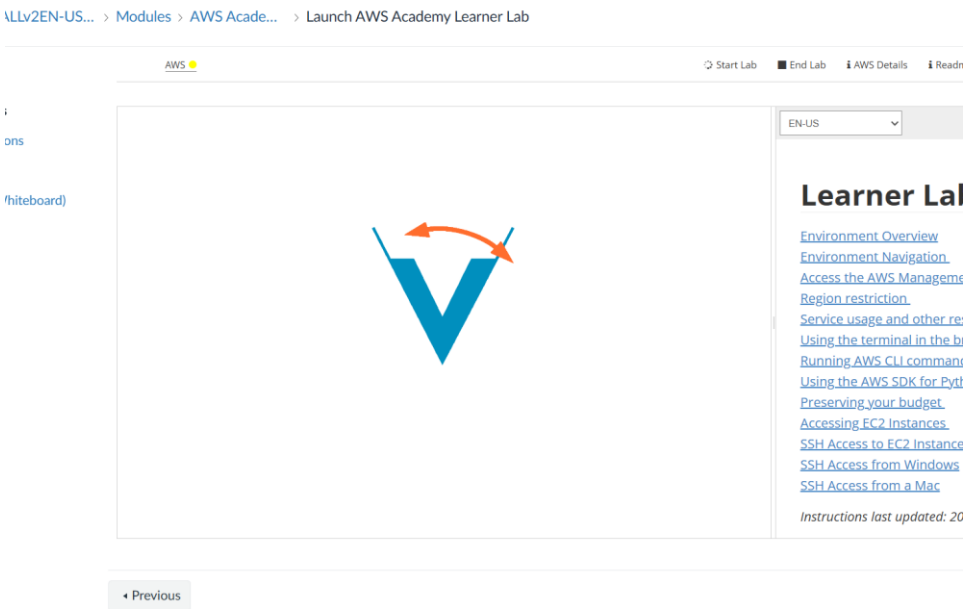
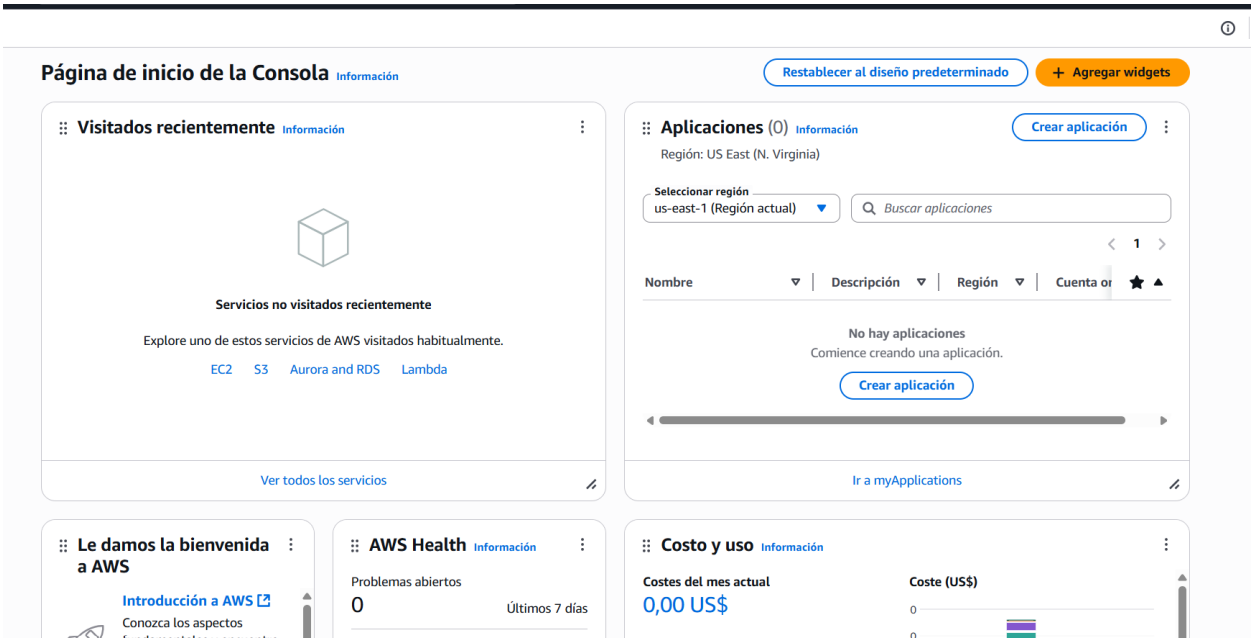


Ilustración 2 Iniciar Laboratorio



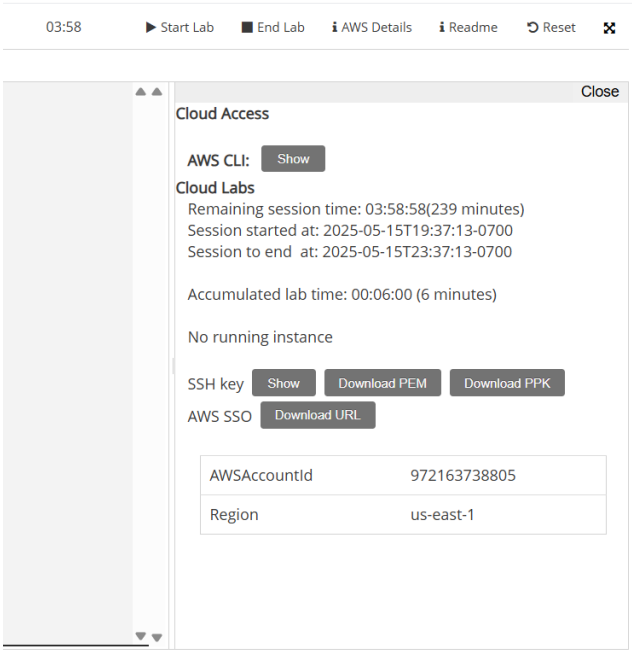
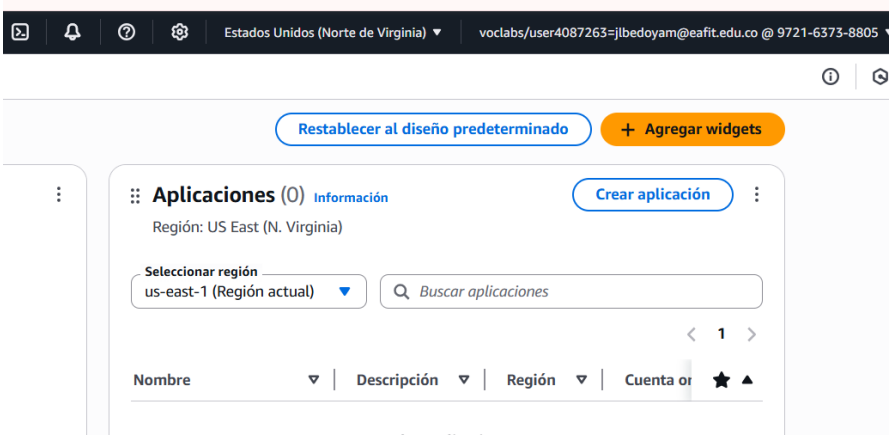


Ilustración 3. Descargar el archivo de credenciales. PEM para conexión remota por SSH



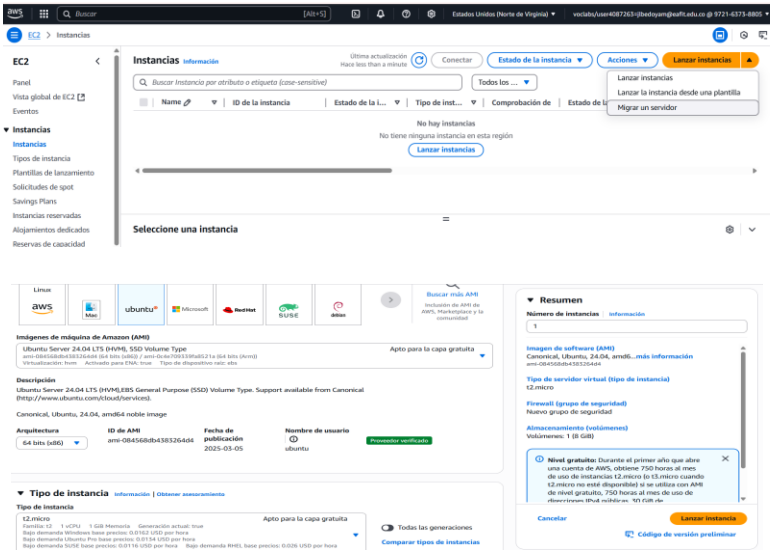


Ilustración 4. Crear una máquina virtual sencilla Ubuntu 22.04

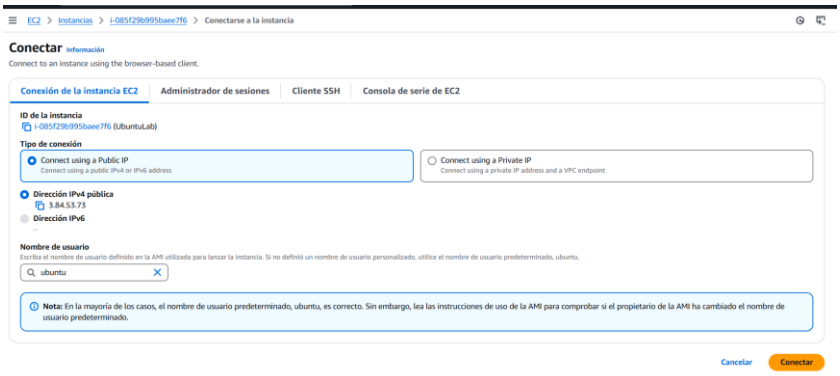


Ilustración 5. verificar que todo funciona correctamente

2. LABORATORIO 1:

ALMACENAMIENTO DE DATOS EN AWS S3

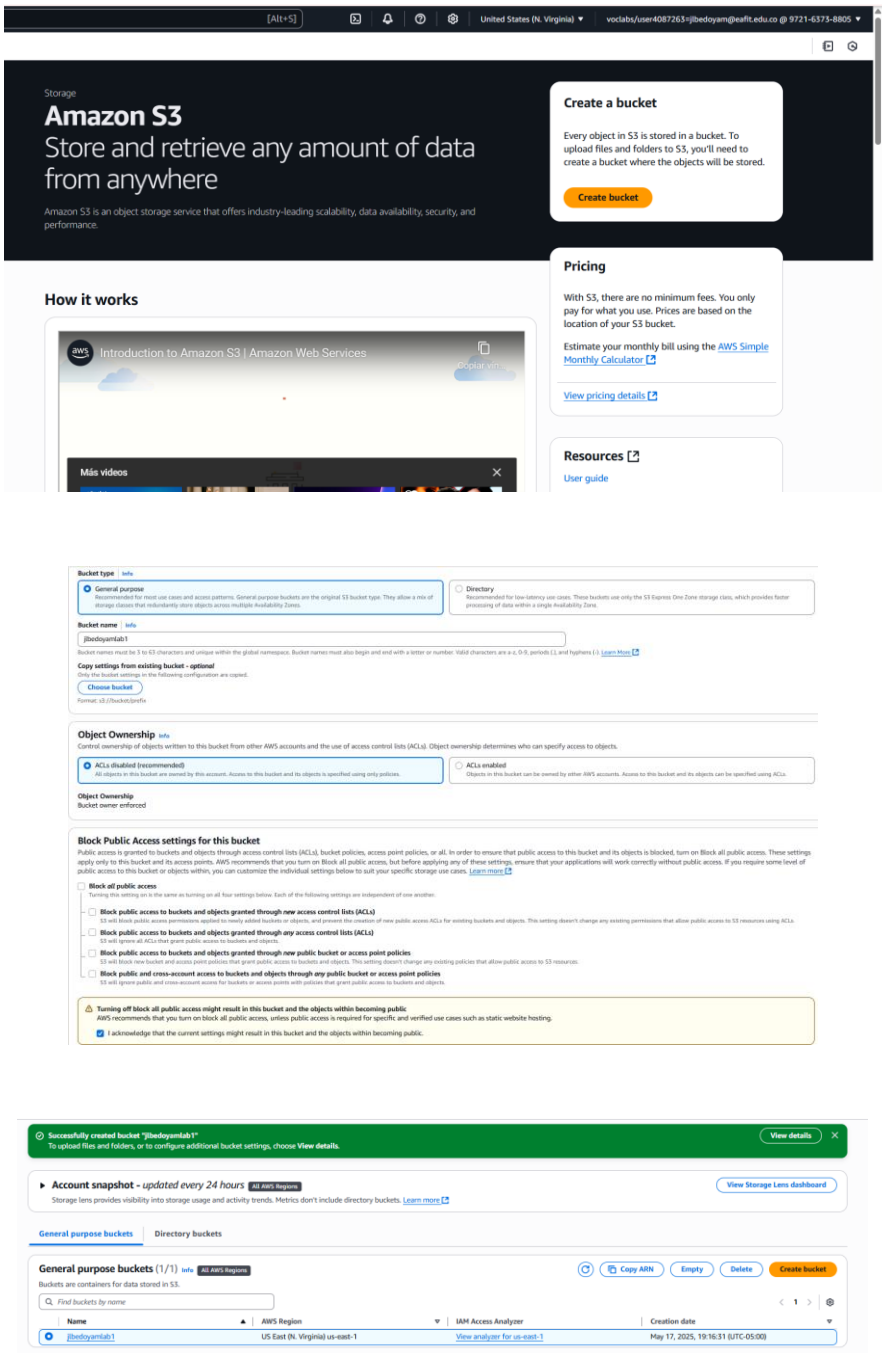


Ilustración 6. Crear Nuevo Bucket

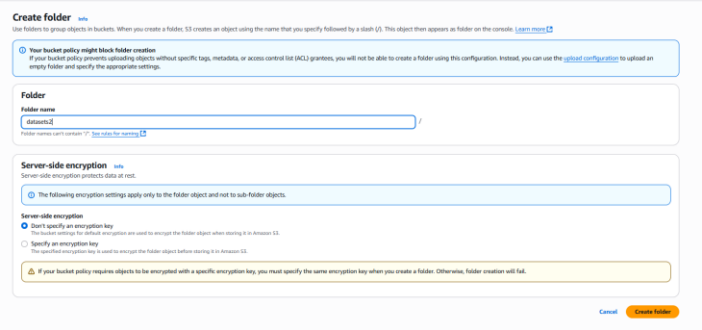
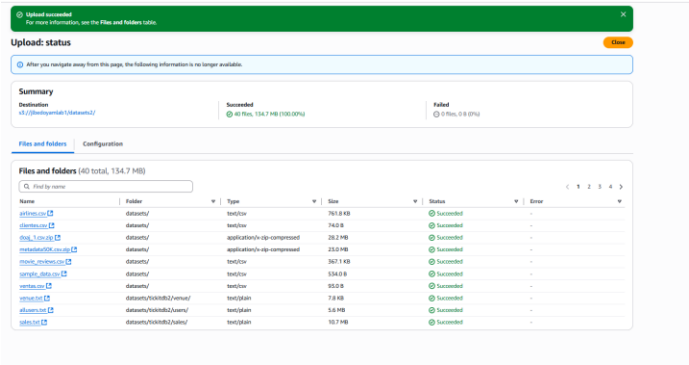
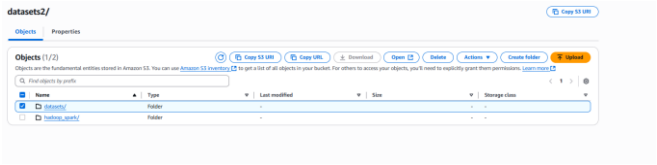


Ilustración 7. Crear Folder





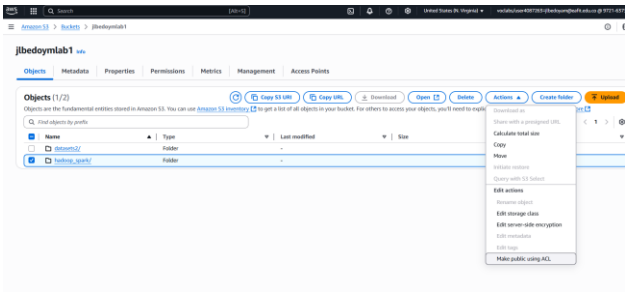


Ilustración 8. Copiar Archivos a Bucket

s3://jlbedoymlab1/datasets/

3. LABORATORIO 2:

IMPLEMENTACIÓN DE UN DATA WAREHOUSE SENCILLO CON AWS ATHENA.

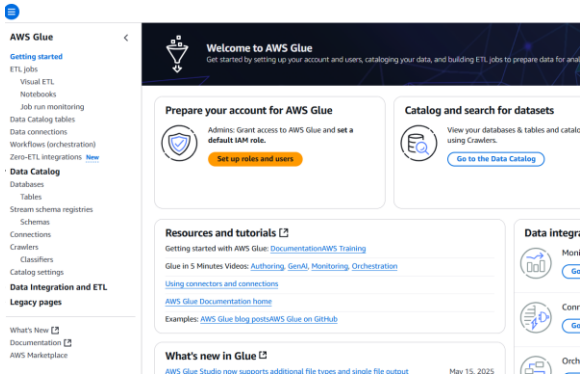
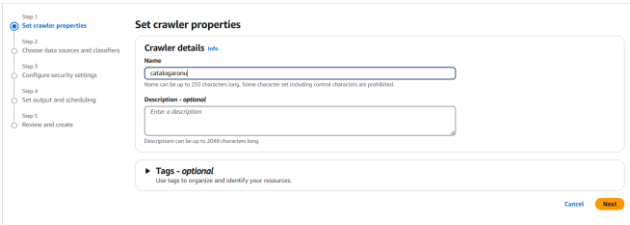


Ilustración 9. Ingreso a AWS Glue



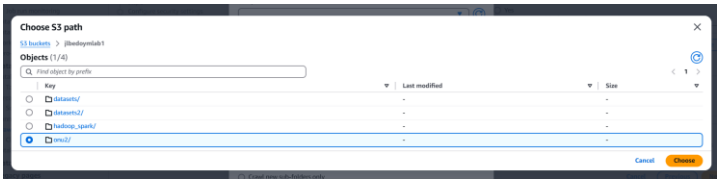


Ilustración 10. Set Crawlers

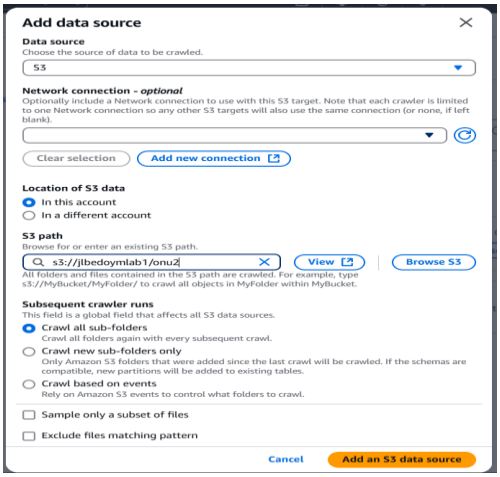


Ilustración 11. Conectar a S3

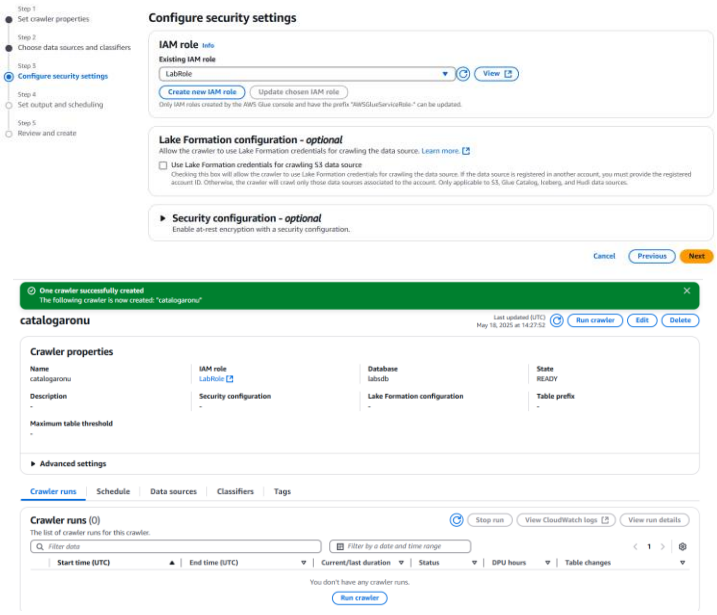


Ilustración 12. Run Crawler

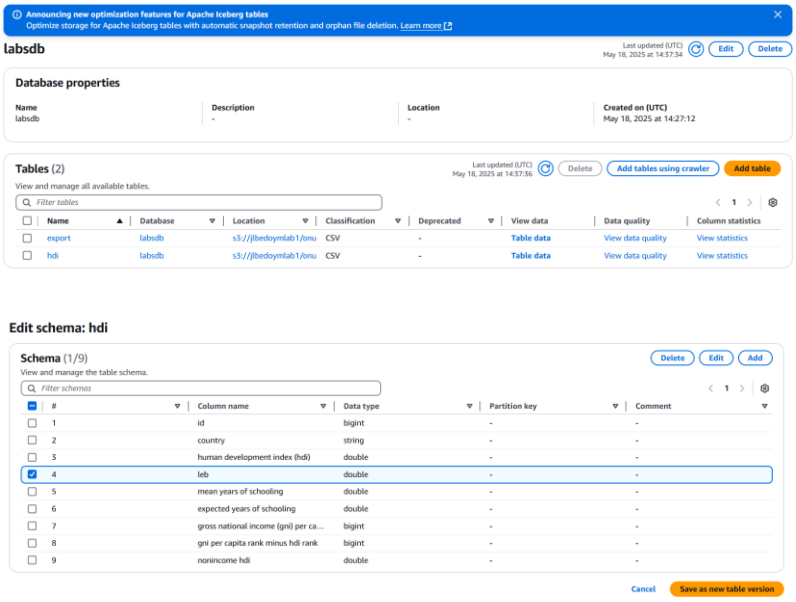


Ilustración 13: Catalogación con AWS Glue

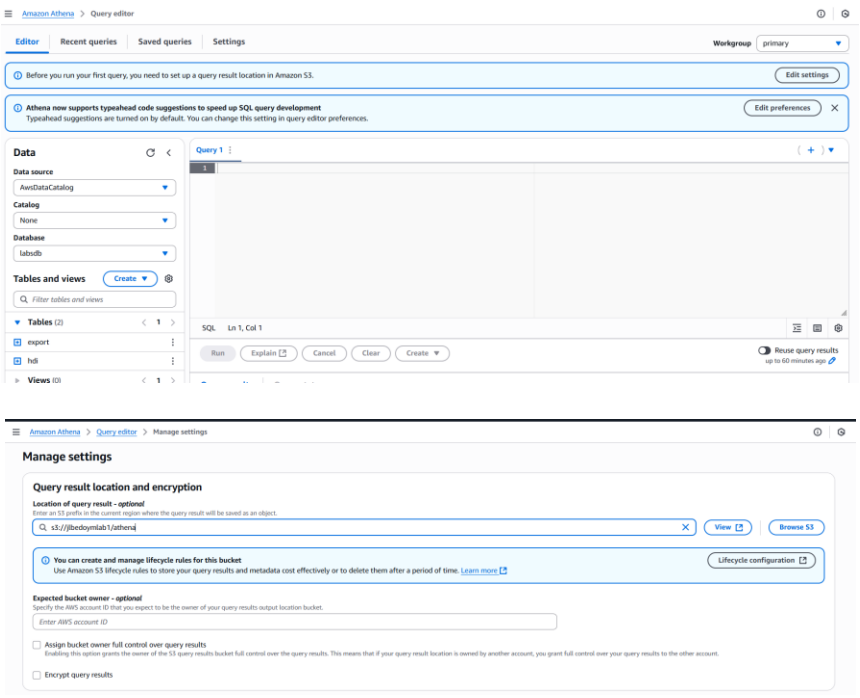
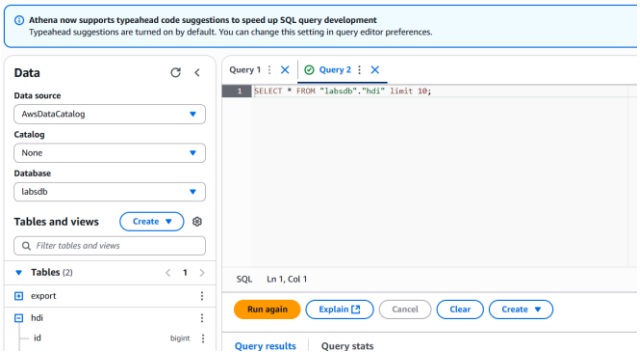


Ilustración 14. configuración Directorio Athena



Crear base de datos tickit (ejercicio propuesto)

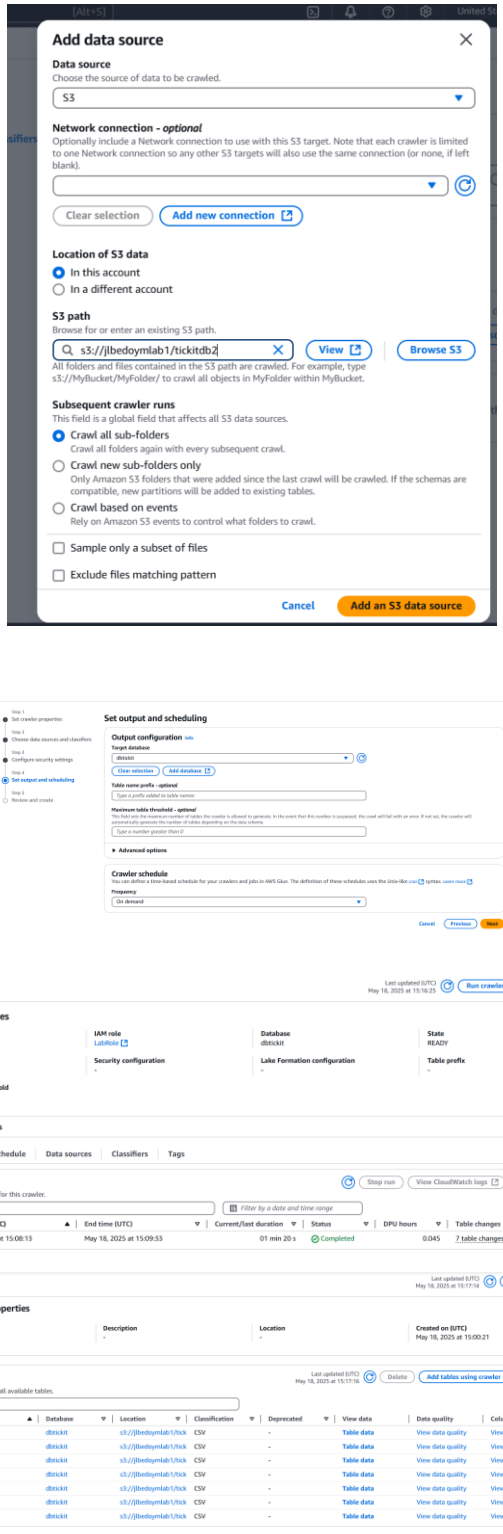


Ilustración 15. Implementación DWH Tickit

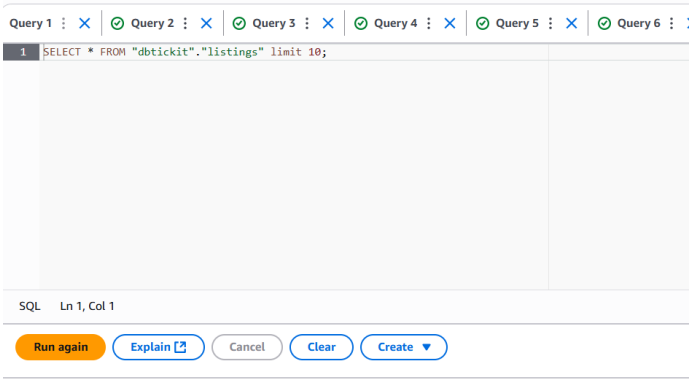


Ilustración 16. Algunas consultas

4. LABORATORIO 3:

IMPLEMENTACIÓN DE UN DATA WAREHOUSE CON AWS REDSHIFT y REDSHIFT SPECTRUM

LAB 3.1: REDSHIFT

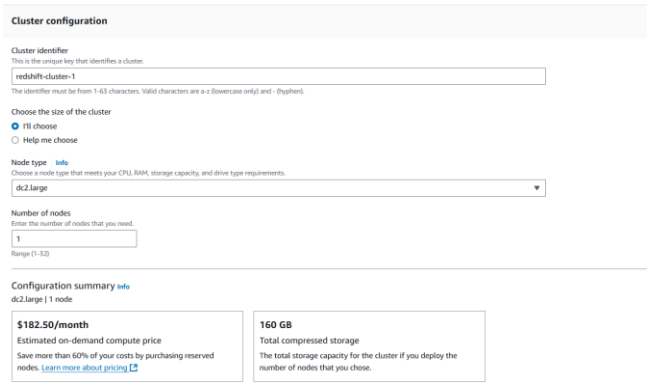
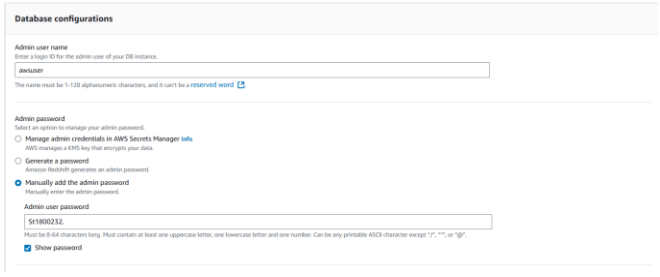


Ilustración 17. Creación Cluster



St1800232.

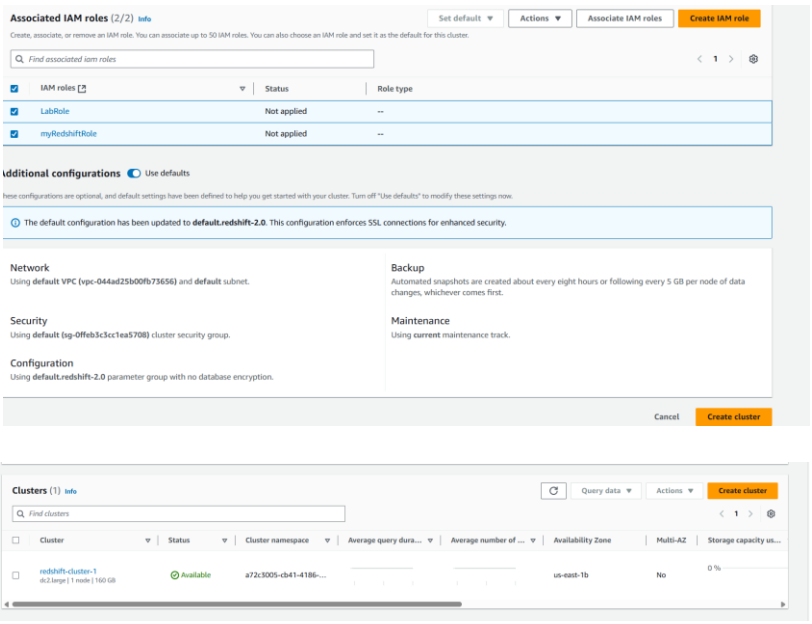
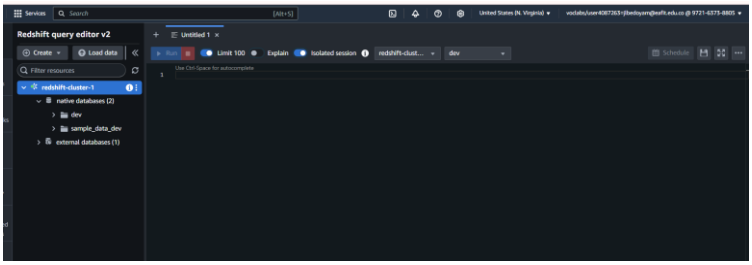
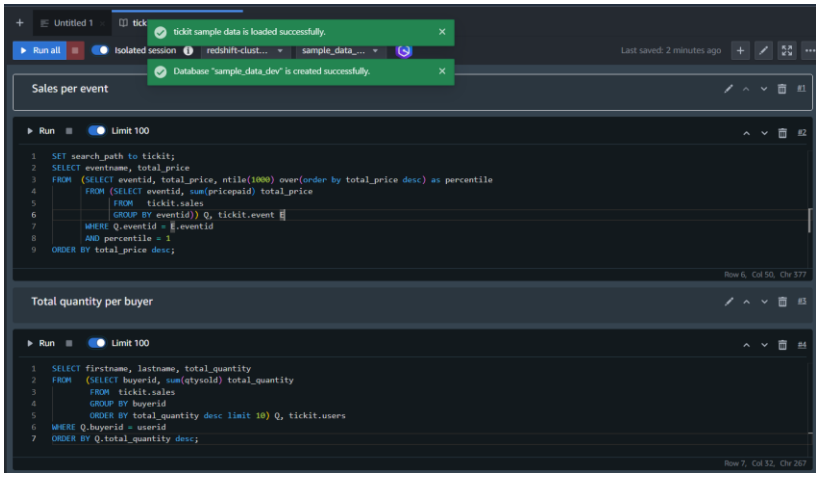


Ilustración 18. configuración Cluster





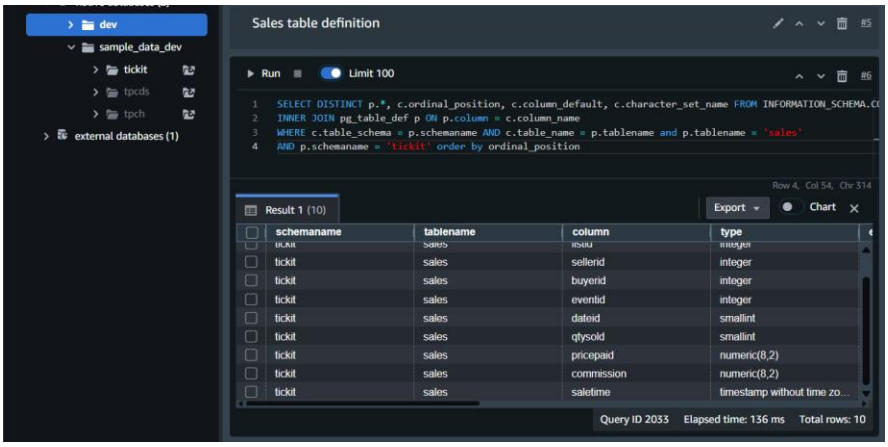
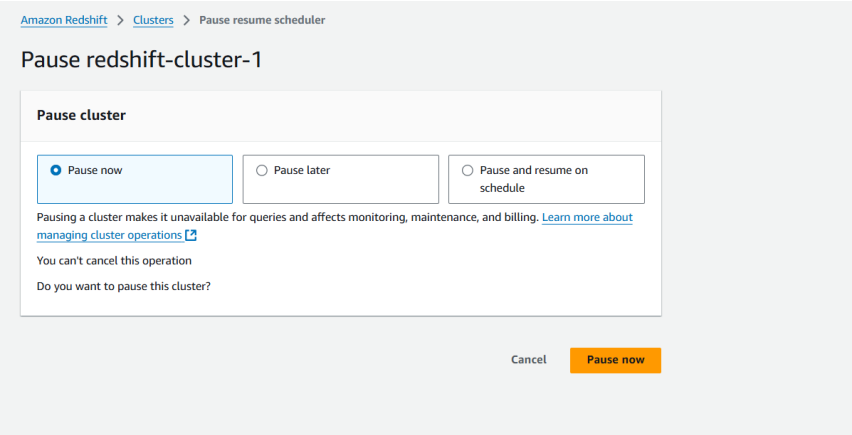
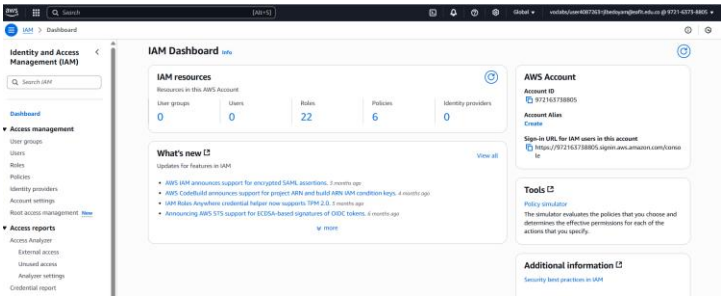


Ilustración 19. Consultas en Redshift



LAB 3.2: RDSHIFT SPECTRUM



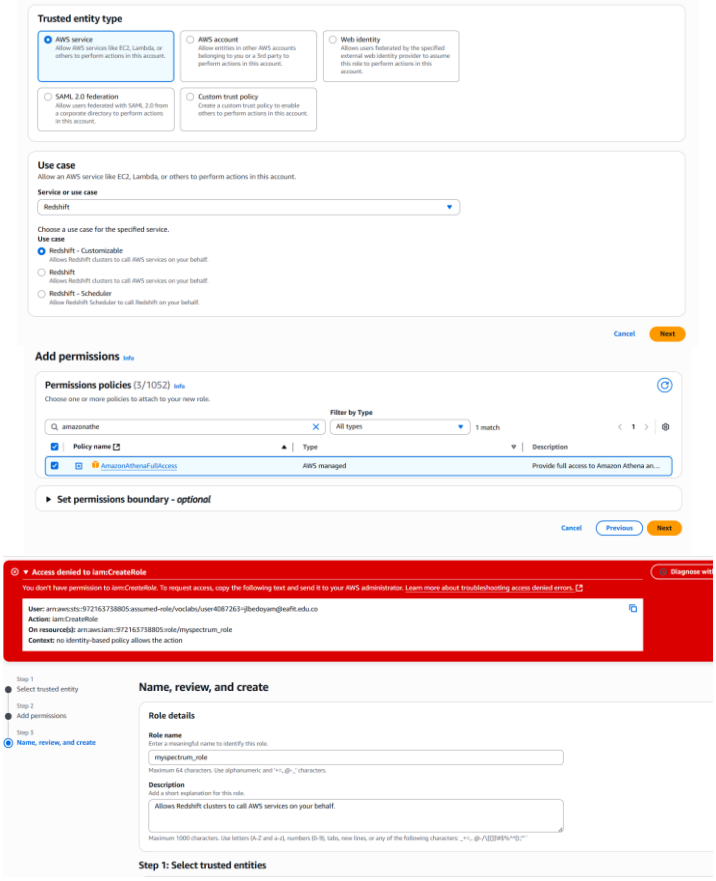


Ilustración 20. Crear un rol IAM para Amazon Redshift

arn:aws:iam::972163738805:role/LabRole

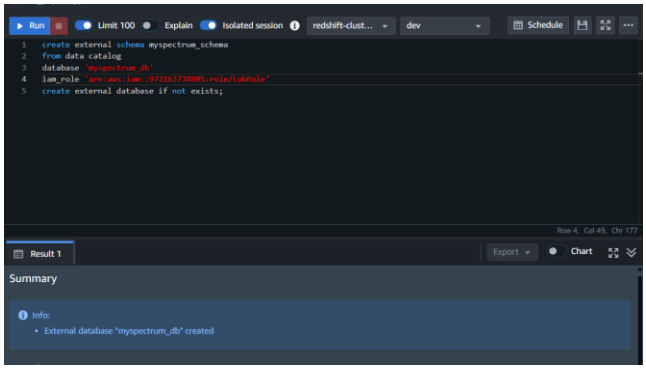


Ilustración 21. Crear tabla externa

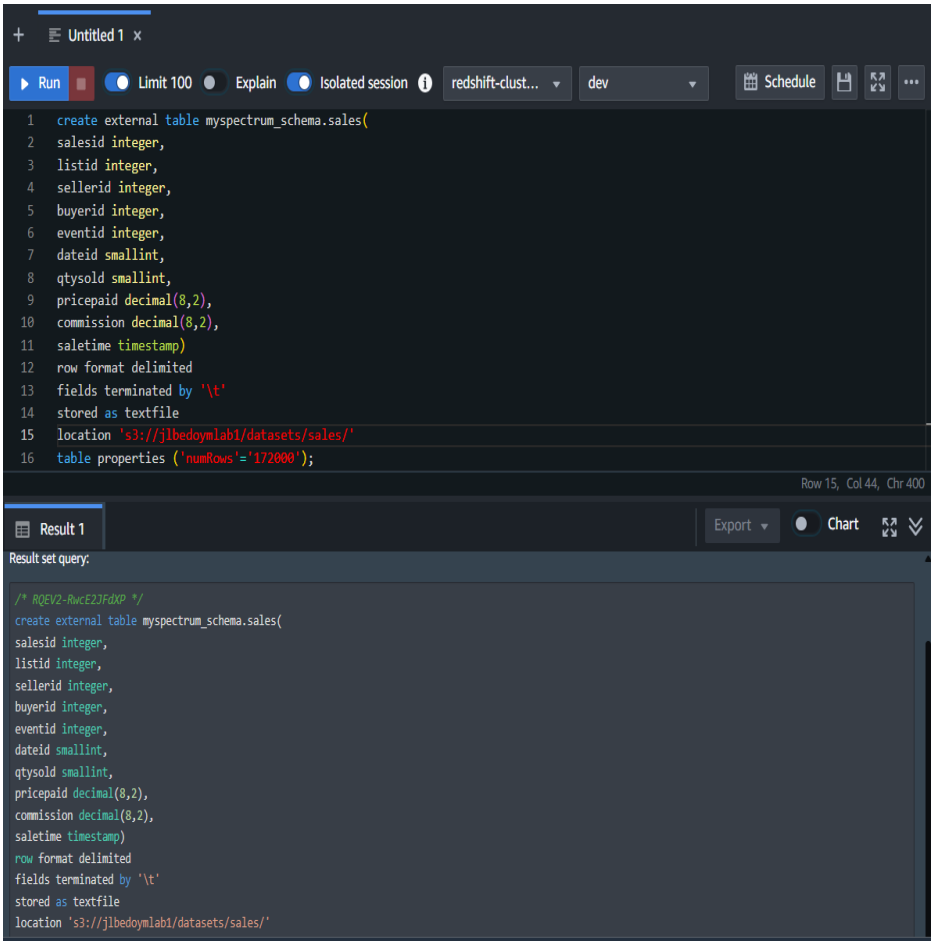


Ilustración 22. Crear una tabla con datos externos en S3

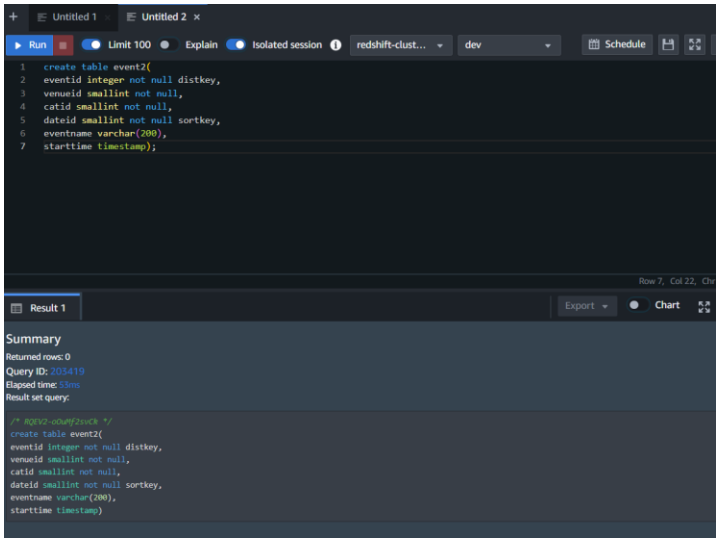


Ilustración 23.Crear una tabla nativa en redshit

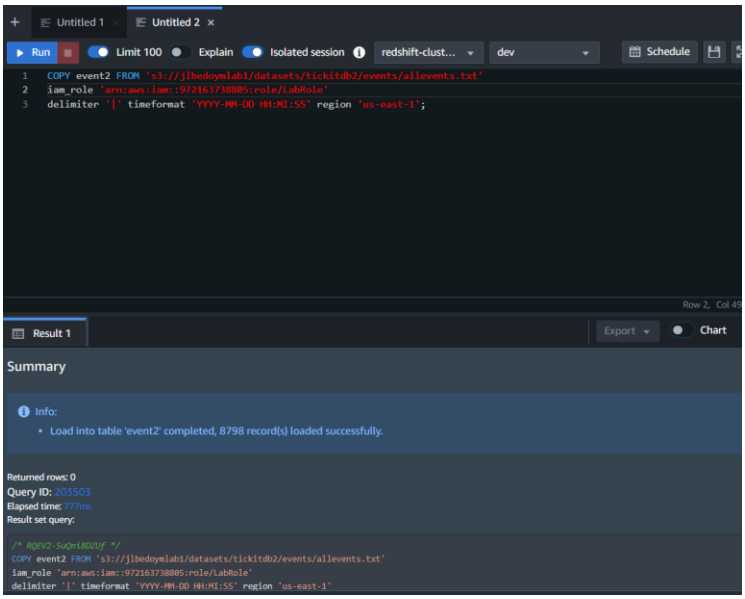


Ilustración 24. Cargar datos en la table 'event2

5. LABORATORIO 4:

AWS REDSHIFT con Aprendizaje de Máquina.

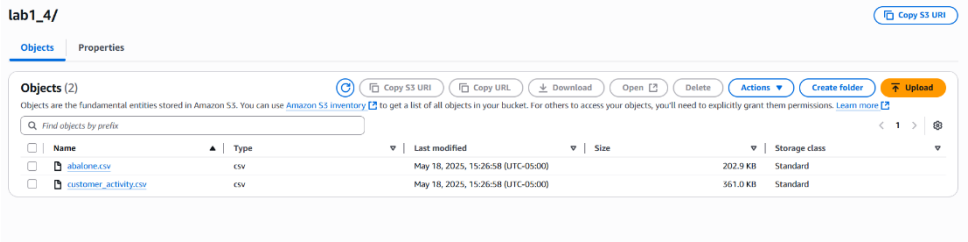


Ilustración 25. Descargar datos y Almacenar en S3

s3://jlbedoymlab1/lab1_4/

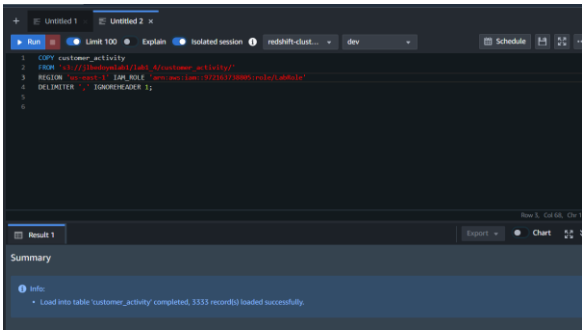


Ilustración 26. Copiar datos desde S3

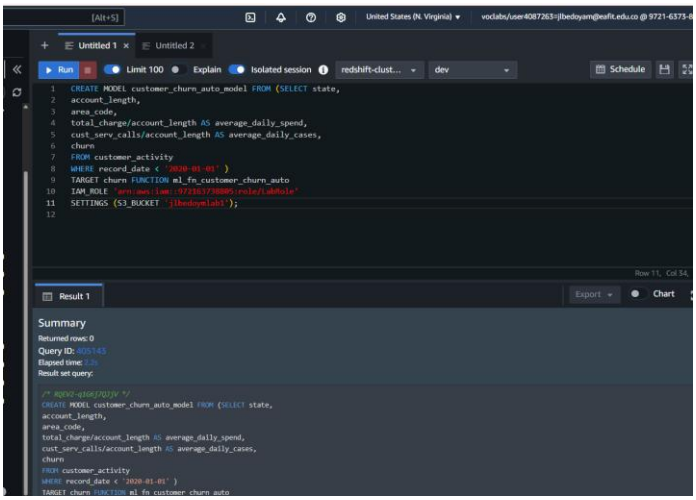


Ilustración 27. Crear Modelo

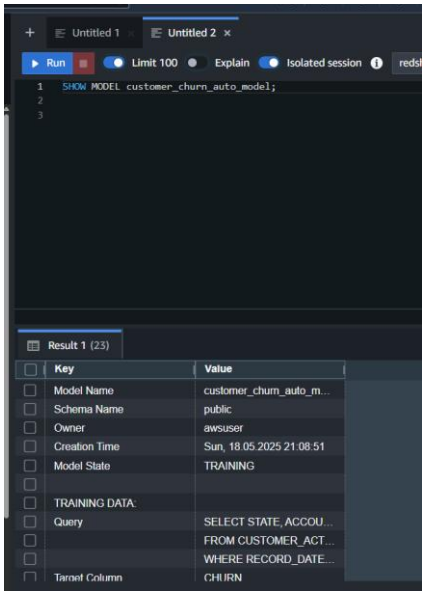
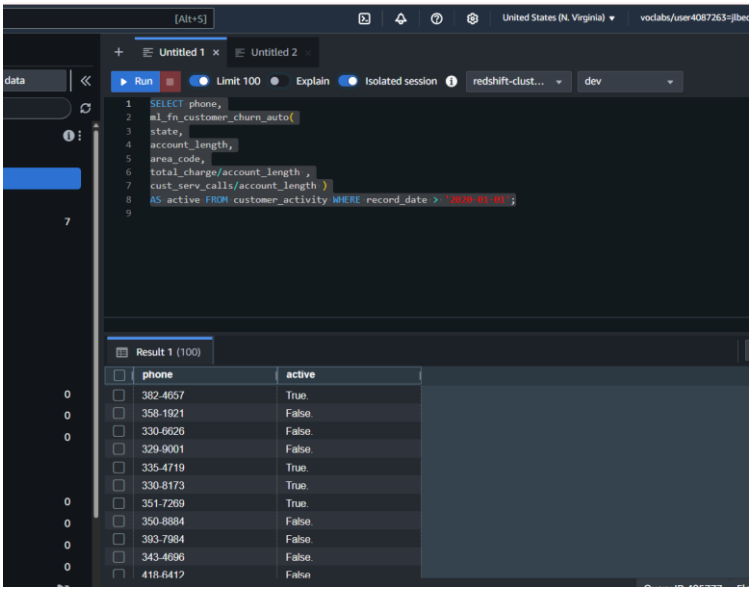


Ilustración 28. Verificar estado del modelo



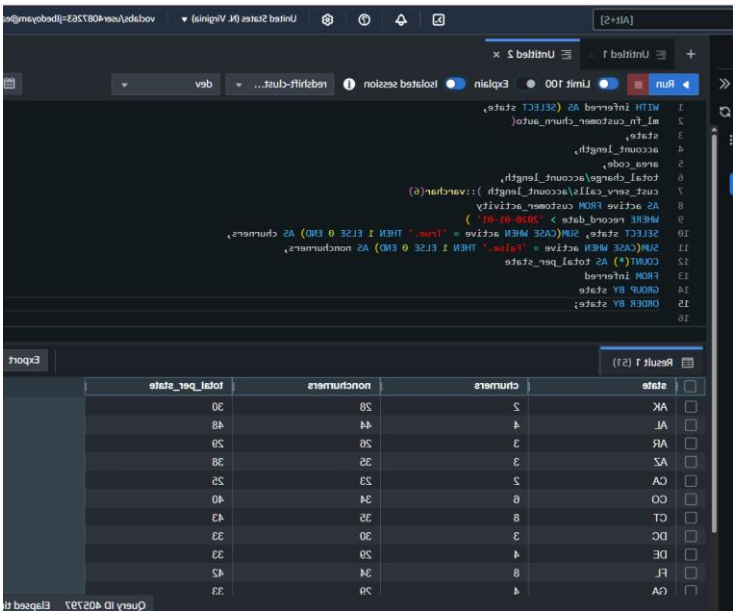


Ilustración 29. Predicciones del modelo

Parte 2:

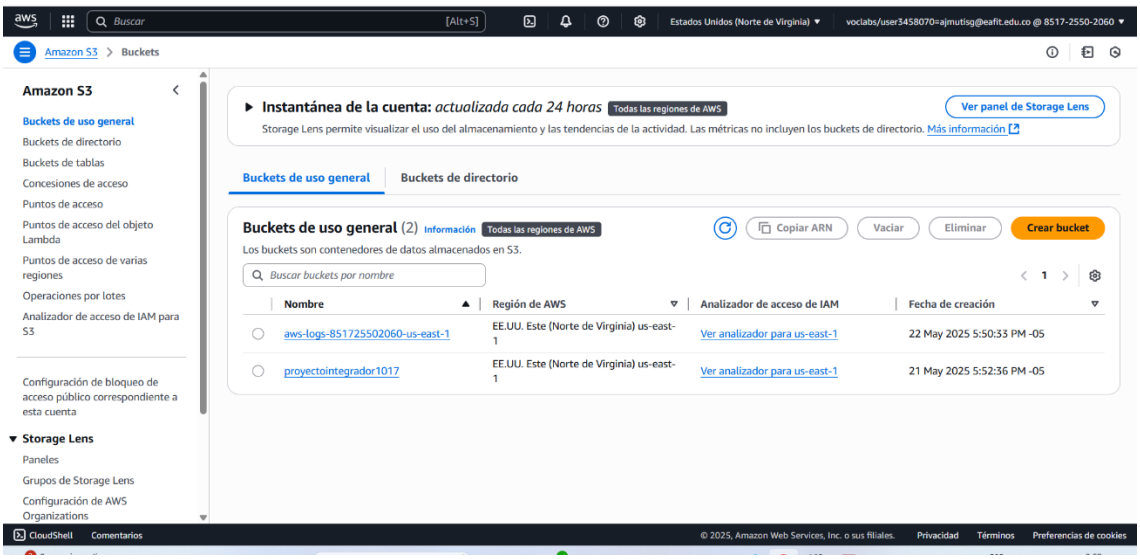
Trabajo 1 – Diseño e implementación de una solución analítica básica en una arquitectura Batch en AWS.

Desarrollar el proyecto basado en un ciclo de vida clásico de la Arquitectura Batch:

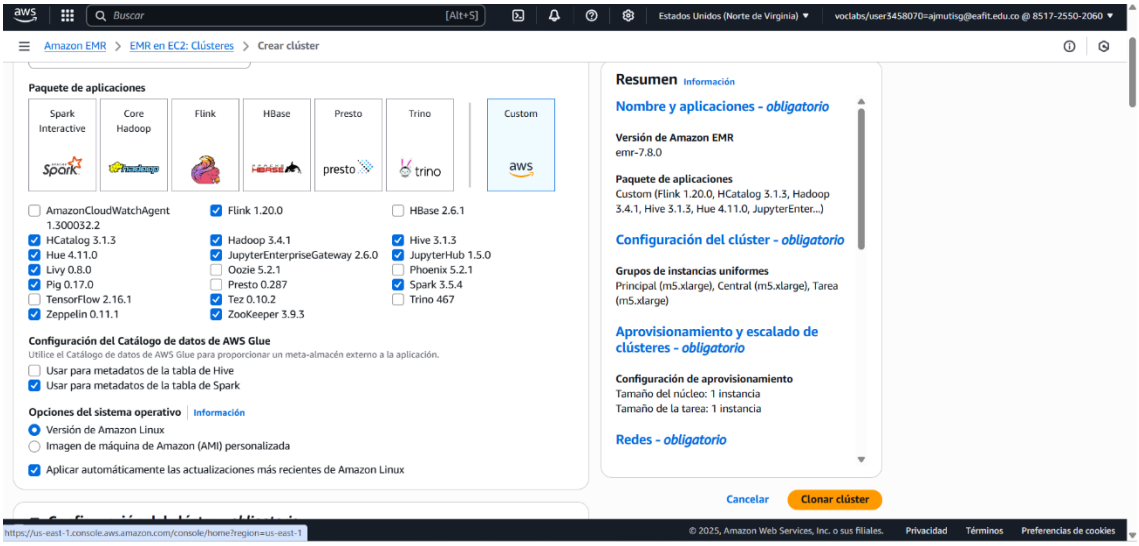
1) Fuentes de datos: Datos del PROYECTO INTEGRADOR.
2) Ingesta de los datos: copia de archivos desde la interfaz web de AWS S3 o desde EMR/S3.
3) Almacenamiento: almacenar los datos originales en la zona raw, los datos preparados en la zona 'trusted' y los datos analizados, así como modelos en la zona 'refined'.
4) Procesamiento de datos:
    a. La preparación de datos, la puede hacer utilizando dataframes de Spark, y deberá colocar los datos preparados en la zona 'trusted'.
    b. Catalogación de datos para EDA con SparkSQL: Realizar la catalogación de datos (creación de los esquemas) con una de 2 herramientas: AWS Glue (automática) o con EMR/Hive (manual). La base de datos para las tablas generadas deberá llamarse 'proyecto1db'
    c. Realizar Análisis Exploratorio de Datos con SparkSQL, utilizando como entrada las tablas de la base de datos 'proyecto1db', los datos de salida deben ser almacenados en la zona 'refined' del datalake/s3.
    d. Diseñar e implementar un modelo básico de Aprendizaje Automático de la librería SparkML o SparkGraphX. Los datos de salida deben ser almacenados en la zona 'refined' del datalake/s3.
    e. Aplicación: inicialmente realizar algun esquema de visualización de datos, conectándose con la zona 'refined', considerada la zona de salida / acceso, para exponer los resultados del proyecto. Cada equipo podría creativamente definir algun otro mecanismo de aplicación en complemento de la visualización (si aplica).

Desarrollo del trabajo

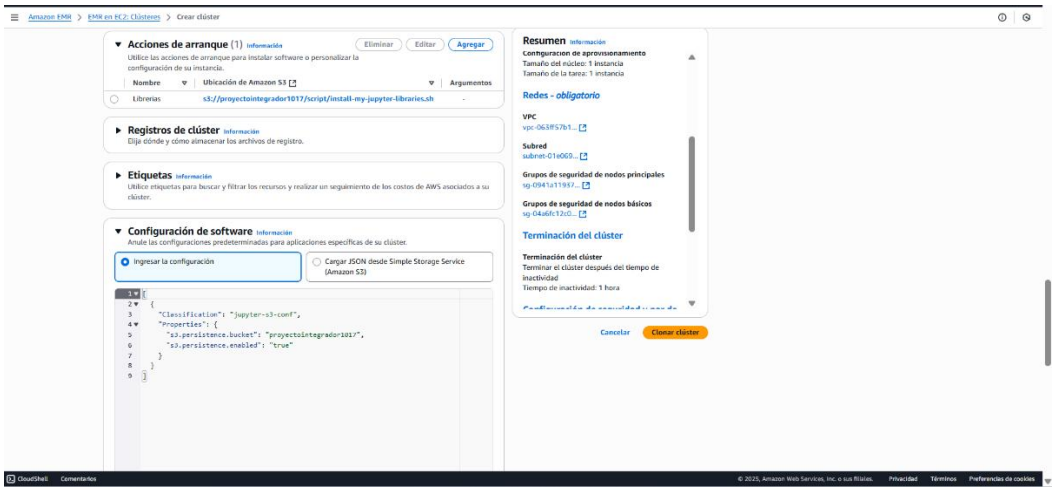
Creación S3 y conexión al cluster



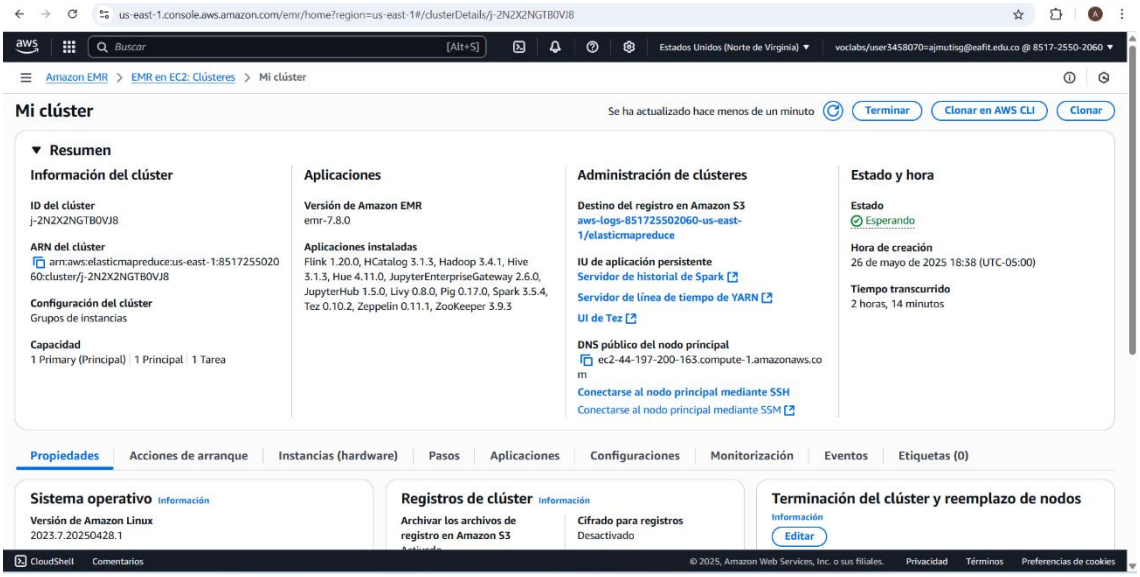
Configuración del cluster



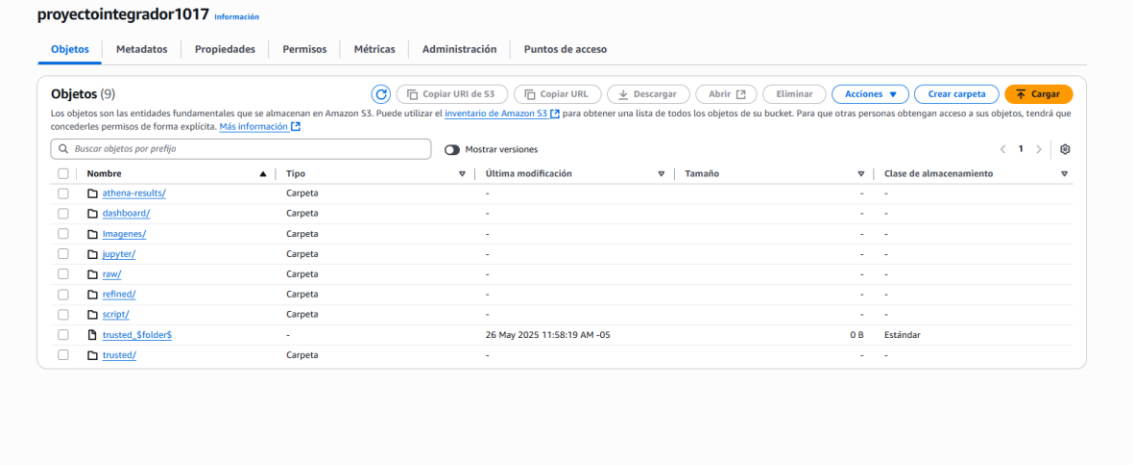
Configuración del cluster



Creación del cluster



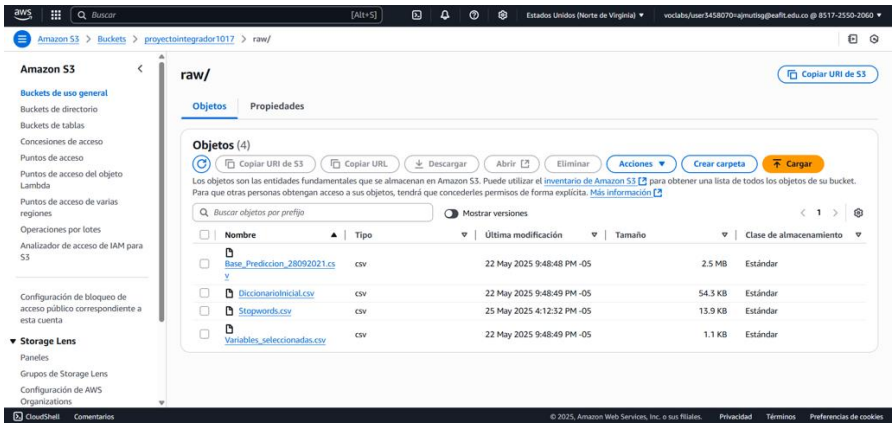
Zona raw, zona trusted y zona refined



Se define la ruta “proyectointegrador1017” en S3 la zona raw para cargar tres archivos CSV en DataFrames:

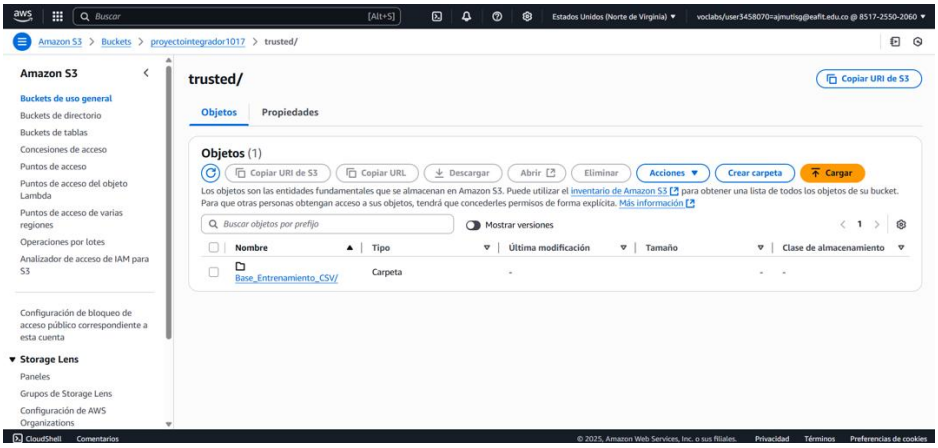
- BasePrediccion_28092021.csv: Contiene los datos a utilizar para el entrenamiento del modelo.
- DiccionarioInicial.csv: Contiene un par de variables para la traducción de palabras similares y la corrección de errores de typeo al ingresar la observación.
- StopWords.csv: Contiene un listado de palabras a ser eliminadas del texto.

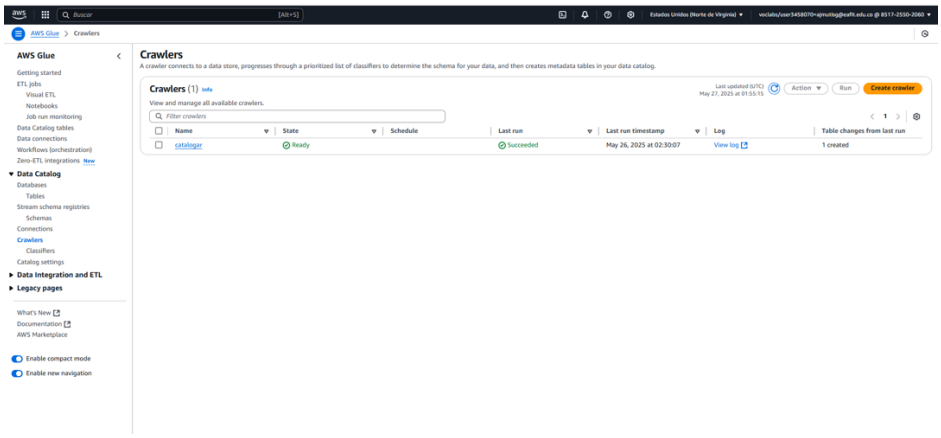
Con esto se constituye la lectura de datos crudos que serán transformados. Posteriormente, se define manualmente un esquema con 30 columnas para asegurar los tipos de variables y que sean consistentes para una vez más leer el archivo BasePrediccion_28092021, pero esta vez con los tipos de variables deseados para evitar errores. Además, es necesario cargar desde S3 la lista de StopWords que serán utilizadas para eliminar el ruido del texto de las observaciones.



Una vez se tiene todo cargado se comienza con la limpieza y depuración del campo observación. Se comienza eliminando el encabezado que contiene información redundante y luego se limpia este fragmento de texto, eliminando caracteres no alfanuméricos, secuencia de números, sellos identificadores e información administrativa de cada registro. Se registran las UDF para cada caso de limpieza y se aplican para generar el df_limpio.

Después, con df_limpio, se toma la lista de stopwords y se crea la UDF para dejar solo las palabras relevantes. También, para normalizar la semántica se aplica un diccionario de sinónimos para estandarizar el vocabulario de las observaciones. A esto se le añadió la codificación de la variable objetivo “Causa Evento” en tres clases: normal, anomalía o no revisado. El resultado de esto es una tabla estructurada con variables categóricas, numéricas y textuales limpias, almacenada en la zona trusted.



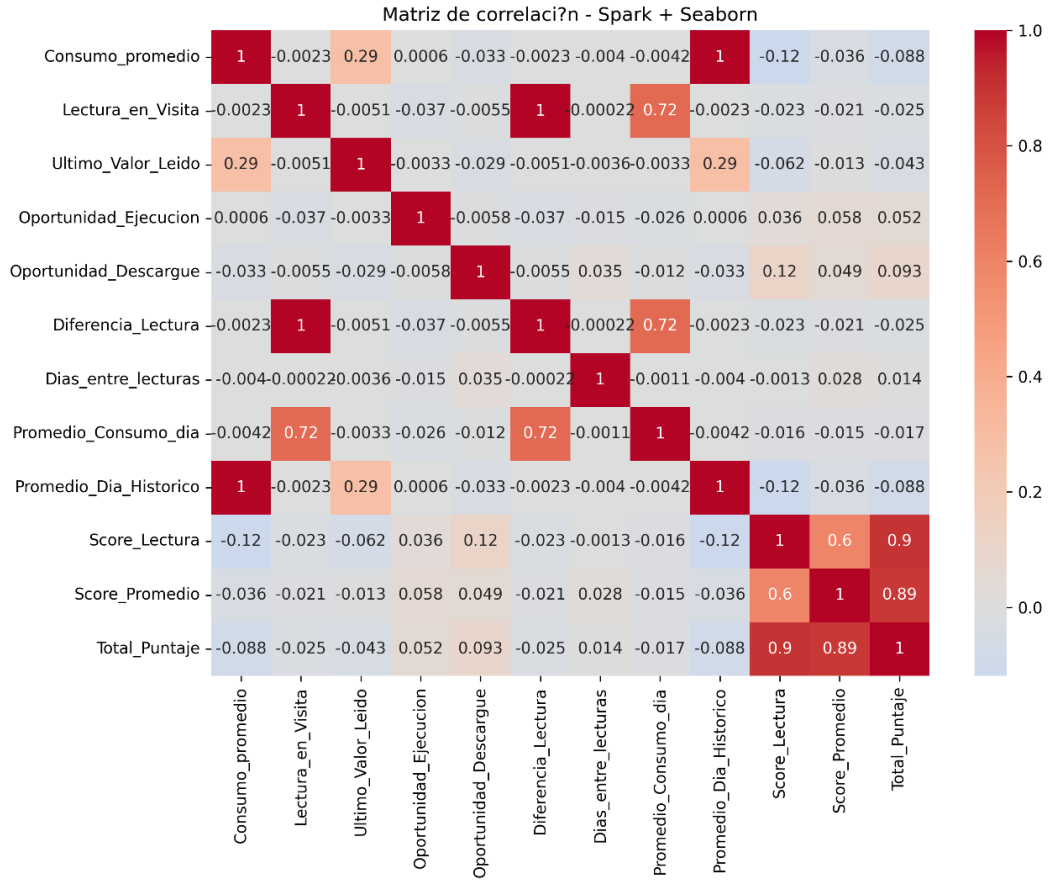


La tabla que se almacenó en la zona trusted se catalogó en AWS Glue y se muestra dentro de la base de datos proyecto1db. Mediante SparkSession habilitado con soporte Hive, se realizan consultas exploratorias con SparkSQL para ver los tipos de columnas, sus nombres exactos y reemplazar algunos nombres para mayor claridad. Se realiza un EDA vía SparkSQL y se almacena en S3. Se sacan estadísticas descriptivas básicas, tablas de frecuencia y correlaciones.

Al analizar la proporción de valores nulos en las doce métricas numéricas evaluadas, se observa que varias presentan niveles de ausencia cercanos al 30 %, lo que demanda decisiones críticas en cuanto a imputación o eliminación. Específicamente, “Consumo_promedio” tiene un 21,3 % de datos faltantes, “Lectura_en_Visita” un 19,1 %, “Último_Valor_Leído” un 11,3 %, “Oportunidad_Ejecución” un 18,5 %, “Oportunidad_Descargue” un 18,6 %, “Diferencia_Lectura” un 28,1 %, “Días_entre_lecturas” un 28,7 %, “Promedio_Consumo_día” un 29,2 %, “Promedio_Día_Histórico” un 21,8 %, “Score_Lectura” un 29,1 %, “Score_Promedio” un 29,2 % y “Total_Puntaje” un 29,2 %. Este nivel de ausentismo sugiere que, para variables con distribuciones sesgadas, se puede imputar usando la mediana, mientras que las filas con múltiples valores faltantes podrían eliminarse si la imputación no garantiza fiabilidad.

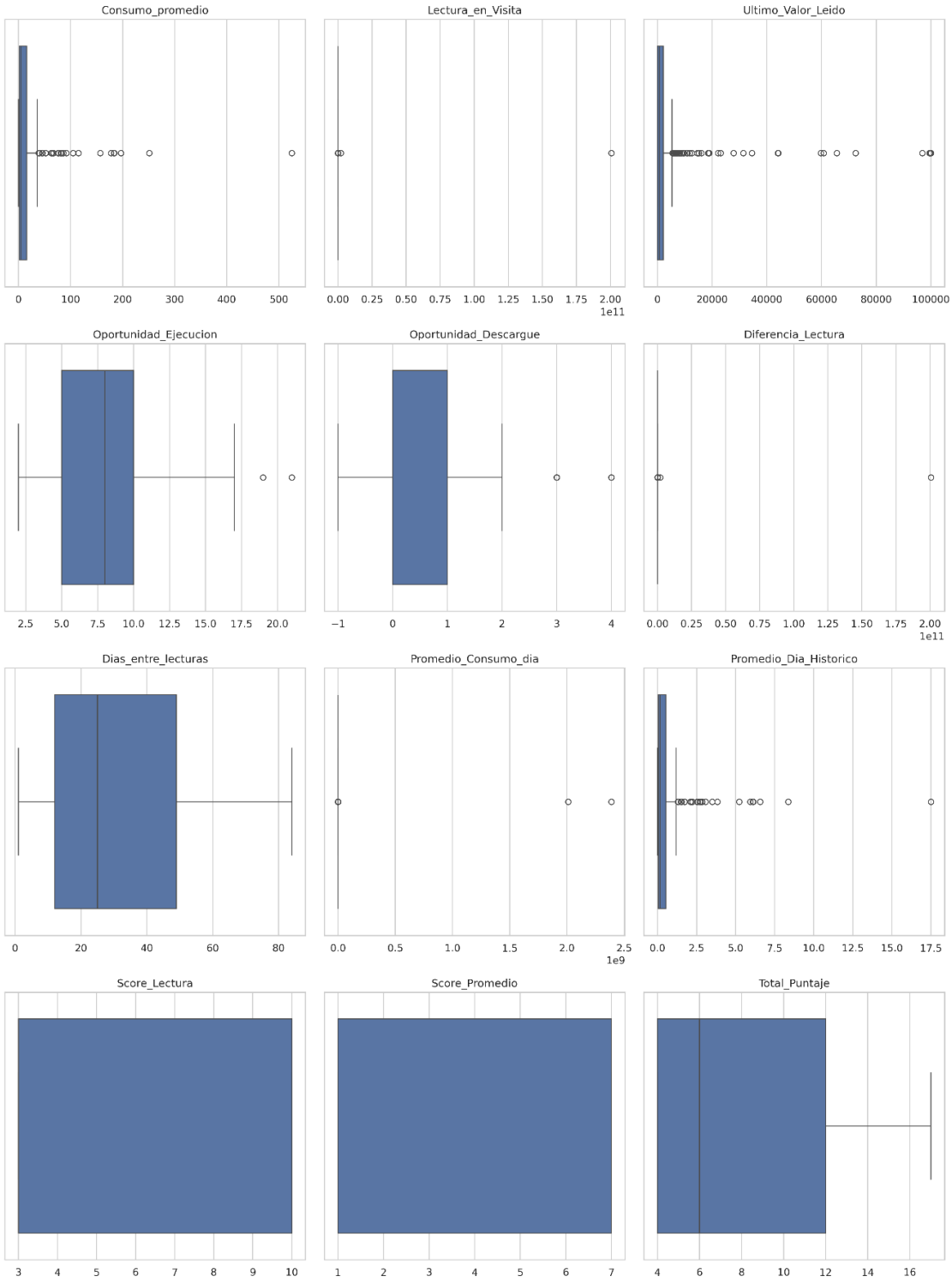
La tabla de estadísticas descriptivas revela que el número de observaciones válidas varía entre 2.514 y 3.148, lo cual refleja las proporciones de datos completos tras la tipificación explícita. “Consumo_promedio” muestra una media de 21,3, desviación estándar de 100,7, y mediana de 5 con cuartiles en [2, 13], indicando que la mayoría de los registros reflejan consumos bajos con presencia de valores extremos. “Lectura_en_Visita” y “Último_Valor_Leído” presentan medias elevadas ($1{,}54\times10^{8}$ y 6.839 respectivamente), con medianas más bajas (392 y 344), lo que

evidencia una distribución sesgada con casos extremos. En cuanto a las variables temporales, “Oportunidad_Ejecución” tiene mediana en 8 días (Q1=5, Q3=10) y “Oportunidad_Descargue” tiende a 0 días en la mayoría de los casos. “Diferencia_Lectura” y “Días_entre_lecturas” exhiben amplitud intercuartílica ([0–34] y [10–42]) con medianas de 5 y 21 días. Finalmente, los indicadores de desempeño —“Score_Lectura”, “Score_Promedio” y “Total_Puntaje”— tienen valores centrales de 3, 1 y 4 respectivamente, lo que sugiere que la mayoría de los casos se concentran en evaluaciones bajas, aunque existen algunos registros con puntajes significativamente altos (hasta 14 en lectura y 17 en puntaje total).



Al inspeccionar el heatmap, Promedio_Dia_Historico y Consumo_promedio muestran correlación prácticamente unitaria, por un lado, Score_Lectura, Score_Promedio y Total_Puntaje forman un bloque muy correlacionado (por encima de 0,8) al aplicar la misma lógica de cálculo. Promedio_Consumo_dia presenta correlaciones moderadas (≈ 0,7) con Lectura_en_Visita y Dias_entre_lecturas, lo que tiene sentido porque más visitas o un intervalo mayor de días suelen

traducirse en consumos diarios superiores; el resto de los pares de variables exhibe correlaciones bajas (por debajo de 0,3), indicando que esas métricas aportan información más independiente.



Al recorrer los boxplots observamos que Consumo_promedio, Promedio_Dia_Historico y Ultimo_Valor_Leido presentan distribuciones fuertemente asimétricas con medianas muy cercanas a cero y multitud de valores atípicos que alcanzan centenas o incluso miles de veces el cuartil superior, lo cual refleja picos excepcionales de consumo o lecturas extraordinarias, las variables de tiempo como Oportunidad_Ejecucion, Oportunidad_Descargue, Diferencia_Lectura y Dias_entre_lecturas muestran rangos intercuartílicos más homogéneos alrededor de sus medianas (8, 0, 5 y 21 días respectivamente) pero también con algunos outliers que indican casos de retrasos o esperas muy prolongadas, Promedio_Consumo_dia mantiene la mayoría de sus observaciones por debajo de 2 unidades diarias, y en los puntajes Score_Lectura, Score_Promedio y Total_Puntaje los boxplots revelan una concentración central entre 3–10, 1–7 y 4–12 con escasos valores fuera de rango, estos patrones de dispersión y presencia de outliers nos dan ya indicios claros de qué variables pueden requerir transformaciones.

Finalizando, se complementa el análisis exploratorio de las variables con eliminación de variables categóricas redundantes mediante pruebas Chi-cuadrado, se eliminan variables numéricas altamente correlacionadas según el coeficiente de Pearson, se aplican funciones personalizadas (UDF) para reclasificar la variable objetivo y se seleccionan las variables finales para consolidad un DataFrame limpio y listo para el entrenamiento del modelo.

También, aplicamos un criterio más laxo de eliminación de nulos, descartando únicamente aquellas filas que tenían todas las variables numéricas en null y, sobre ese resultado, eliminando las filas que mostraban todas las variables categóricas nulas o vacías. Los resultados fueron los siguientes, partiendo de 3 550 registros:

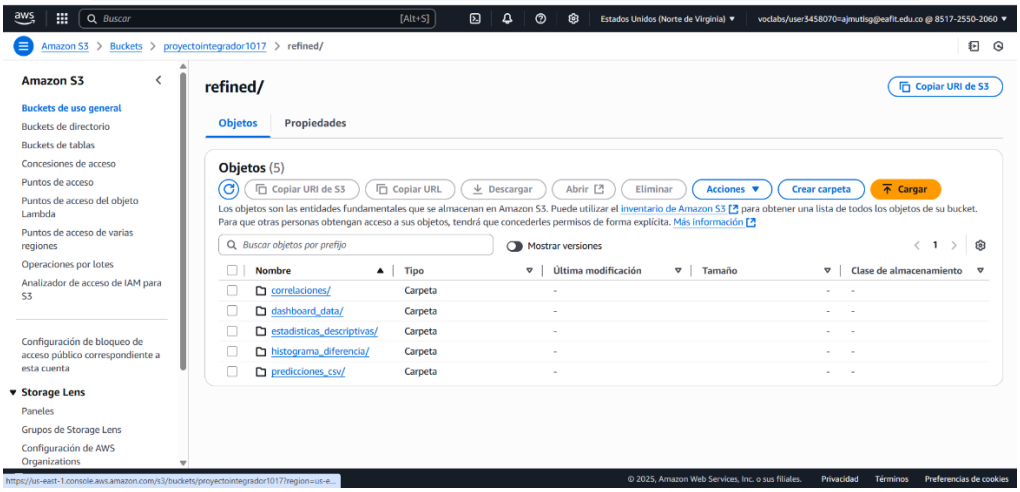
- Tras eliminar filas con todas las numéricas nulas quedaron 3 150 registros, es decir, se redujo el 11,3 % del total
- Al aplicar luego el filtro de todas las categóricas nulas o vacías no se eliminó ningún registro adicional, manteniéndose en 3 150 casos
- En conjunto conservamos el 88,7 % de los datos originales, lo que confirma que son pocos los casos con ausencia completa tanto de métricas como de categorías

Dado lo anterior, se entrenó un modelo de clasificación con LogisticRegression, integrado en un pipeline de SparkML conformado por múltiples etapas de preparación de datos y clasificación supervisada. Inicialmente, las variables categóricas fueron indexadas y codificadas en vectores binarios, mientras que las variables numéricas fueron escaladas para normalizar su rango. Paralelamente, el texto preprocesado fue convertido en vectores utilizando bigramas y CountVectorizer. Todas estas características fueron ensambladas en un solo vector unificado que sirvió como entrada para un modelo de regresión logística multiclase. La división del conjunto en entrenamiento (80%) y prueba (20%) permitió evaluar el rendimiento del modelo utilizando métricas de exactitud (accuracy) y F1 score.

Como resultado, el modelo entrenado tuvo valores de 0.8964 de exactitud (accuracy) y 0.8149 en F1 score. Los resultados son predicciones en el que según el texto observado se cataloga como anomalía o normal. Los resultados obtenidos en las métricas de evaluación (precisión y F1 score) reflejan que el modelo logra un desempeño adecuado, lo cual valida la correcta selección y preparación de las variables utilizadas. En particular, el uso conjunto de variables numéricas y categóricas permitió complementar la información objetiva, derivada de registros operativos como consumos, tiempos o puntajes, con dimensiones más cualitativas como la actividad económica, que contextualizan el comportamiento observado.

A pesar de tratarse de un modelo relativamente sencillo como la regresión logística, los resultados muestran que es capaz de predecir con precisión si la clasificación realizada por el operador (evento normal, anómalo o no revisado) es coherente con los patrones subyacentes en los datos. Esto demuestra que, incluso con modelos lineales, es posible capturar relaciones significativas cuando la calidad del dataset ha sido cuidada. Además, existe un margen claro de mejora: una depuración más rigurosa de las observaciones, la ampliación del diccionario de sinónimos o la incorporación de nuevas variables semánticas podrían enriquecer la capacidad predictiva del modelo. Este caso ilustra claramente el valor de combinar datos estructurados y no estructurados dentro de arquitecturas tipo datalake, alineándose con los principios de la analítica moderna orientada a la explotación integral de múltiples fuentes de información para la toma de decisiones basada en datos.

Por último, el resultado es un DataFrame con las predicciones al que se le limpian las columnas innecesarias y se exporta a la zona refined del datalake en S3 “s3://proyectointegrador1017/refined/predicciones_csv/”.



La visualización de resultados se desarrolló a partir de los datos procesados y almacenados en la zona refined, utilizando Plotly para generar un tablero interactivo en formato HTML. Este dashboard incluye tres gráficos clave: (1) un gráfico de barras que muestra la distribución de las categorías predichas (evento_cat), permitiendo verificar visualmente el balance de clases y la capacidad del modelo para distinguir entre eventos normales, anómalos y no revisados; (2) un gráfico de barras que presenta los cinco eventos más frecuentes según el campo evento, lo cual ayuda a identificar patrones recurrentes en el etiquetado manual de los operadores; y (3) un histograma de la longitud de las observaciones textuales (`Observacion_Ok_str`), útil para analizar la riqueza semántica y variabilidad del contenido textual. Este panel se empaquetó como un archivo HTML y fue publicado en S3 bajo la ruta “`s3://proyectointegrador1017/refined/dashboard/dashboard.html`”, ofreciendo un canal práctico para explorar los resultados del modelo en tiempo real y facilitar la toma de decisiones basada en datos.

Dashboard de Predicciones (Refined)